This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Previously Presented) A method of allocating a memory address space to a plurality of Peripheral Component Interconnect (PCI) adapters coupled to a plurality of slots in a PCI bus, the method comprising:

accessing configuration data associated with a slot identifier for each of the plurality of slots to determine a memory range size associated with each slot; and

non-uniformly allocating memory address ranges to the plurality of PCI adapters based upon the memory range sizes associated with each slot, such that first and second PCI adapters among the plurality of PCI adapters that have the same connector type are allocated differently sized memory ranges.

2. (Currently Amended) A method of allocating memory addresses to a plurality of input/output (IO) resources coupled to a plurality of IO endpoints in a memory mapped IO fabric, the method comprising:

determining a location in the memory mapped IO fabric for each IO endpoint among the plurality of endpoints; and

non-uniformly allocating memory address ranges to the plurality of IO endpoints based upon the determined locations of the IO endpoints in the memory mapped IO fabric, wherein allocating memory address ranges to the plurality of IO endpoints includes allocating differently sized memory address ranges to first and second IO endpoints having the same connector type.

3. (Original) The method of claim 2, wherein determining the location and non-uniformly allocating memory address ranges are performed during initialization of the memory mapped IO fabric.

4. (Original) The method of claim 3, wherein determining the location and non-uniformly allocating memory address ranges are performed during initialization of a computer to which the memory mapped IO fabric is coupled.

5. (Original) The method of claim 4, wherein the computer comprises a logically-partitioned computer, and wherein determining the location and non-uniformly allocating memory address ranges are performed by a partition manager in the logically-partitioned computer.

6. (Canceled).

7. (Currently Amended) The method of claim 2 6, wherein the first and second IO endpoints each comprise an IO slot, and wherein the connector types of the first and second IO endpoints have the same data bus width.

8. (Original) The method of claim 2, wherein the memory mapped IO fabric comprises a PCI-compatible fabric.

9. (Original) The method of claim 8, wherein the memory mapped IO fabric comprises at least one PCI-compatible bus, wherein at least a subset of IO endpoints are IO slots coupled to the PCI-compatible bus, and wherein the location of each IO slot is defined by a slot identifier for such IO slot on the PCI-compatible bus.

10. (Original) The method of claim 8, wherein the memory mapped IO fabric comprises a plurality of PCI-compatible buses, wherein at least a subset of IO endpoints are IO slots coupled to the plurality of PCI-compatible buses, and wherein the location of each IO slot is defined by a bus identifier for the PCI-compatible bus to which such IO slot is coupled, and a slot identifier for such IO slot on the PCI-compatible bus to which such IO slot is coupled.

11. (Original) The method of claim 10, wherein the memory mapped IO fabric comprises a plurality of IO enclosures, each IO enclosure including at least one PCI-compatible bus among the plurality of PCI-compatible buses, wherein determining the location of an IO endpoint comprises accessing configuration data associated with the IO enclosure within which such IO endpoint is disposed.

12. (Original) The method of claim 11, wherein non-uniformly allocating memory address ranges to the plurality of IO slots includes allocating a remaining memory address range to a last IO slot among the plurality of IO slots, wherein the remaining memory address range comprises that portion of the memory address range allocated to the PCI-compatible host that was not allocated to each other IO slot among the plurality of IO slots.

13. (Original) The method of claim 12, wherein non-uniformly allocating memory address ranges to the plurality of IO slots includes allocating a remaining memory address range to a last IO slot among the plurality of IO slots, wherein the remaining memory address range comprises that portion of the memory address range allocated to the PCI-compatible host that was not allocated to each other IO slot among the plurality of IO slots.

14. (Original) The method of claim 2, wherein the IO fabric includes at least one IO fabric element allowing connectivity to a subset of the plurality of IO endpoints, and wherein the memory address ranges that would be allocated to IO resources coupled to each IO endpoint in the plurality of IO endpoints are determinable from a publication available prior to installation of IO resources in the subset of IO endpoints.

15. (Currently Amended) An apparatus, comprising:

at least one processor;

program code configured to be executed by the at least one processor to allocate memory address space to a plurality of input/output (IO) resources

coupled to a plurality of IO endpoints in a memory mapped IO fabric by determining a location in the memory mapped IO fabric for each IO endpoint among the plurality of endpoints, and non-uniformly allocating memory address ranges to the plurality of IO endpoints based upon the determined locations of the IO endpoints in the memory mapped IO fabric, wherein the program code is configured to allocate memory address ranges to the plurality of IO endpoints by allocating differently sized memory address ranges to first and second IO endpoints having the same connector type.

16. (Original) The apparatus of claim 15, wherein the program code is configured to determine the location and non-uniformly allocate memory address ranges during initialization of the memory mapped IO fabric.

17. (Original) The apparatus of claim 16, wherein the program code is configured to determine the location and non-uniformly allocate memory address ranges during initialization of the apparatus.

18. (Original) The apparatus of claim 17, wherein the apparatus comprises a logically-partitioned computer, and wherein the program code is resident in a partition manager in the logically-partitioned computer.

19. (Canceled).

20. (Currently Amended) The apparatus of claim 15 ~~19~~, wherein the first and second IO endpoints each comprise an IO slot, and wherein the connector types of the first and second IO endpoints have the same data bus width.

21. (Original) The apparatus of claim 15, wherein the memory mapped IO fabric comprises a PCI-compatible fabric.

22. (Original) The apparatus of claim 21, wherein the memory mapped IO fabric comprises at least one PCI-compatible bus, wherein at least a subset of IO endpoints are IO slots coupled to the PCI-compatible bus, and wherein the location of each IO slot is defined by a slot identifier for such IO slot on the PCI-compatible bus.

23. (Original) The apparatus of claim 21, wherein the memory mapped IO fabric comprises a plurality of PCI-compatible buses, wherein at least a subset of IO endpoints are IO slots coupled to the plurality of PCI-compatible buses, and wherein the location of each IO slot is defined by a bus identifier for the PCI-compatible bus to which such IO slot is coupled, and a slot identifier for such IO slot on the PCI-compatible bus to which such IO slot is coupled.

24. (Original) The apparatus of claim 23, wherein the memory mapped IO fabric comprises a plurality of IO enclosures, each IO enclosure including at least one PCI-compatible bus among the plurality of PCI-compatible buses, wherein the program code is configured to determine the location of an IO endpoint by accessing configuration data associated with the IO enclosure within which such IO endpoint is disposed.

25. (Original) The apparatus of claim 24, wherein the configuration data associated with a first IO enclosure among the plurality of IO enclosures is stored in non-volatile memory resident in the first IO enclosure.

26. (Original) The apparatus of claim 23, wherein a PCI-compatible bus among the plurality of PCI-compatible buses includes a PCI-compatible host bridge, wherein the program code is further configured to allocate a memory address range to the PCI-compatible host bridge, and wherein the program code is configured to non-uniformly allocate memory address ranges to the plurality of IO endpoints by allocating memory address ranges to a plurality of IO slots coupled to the PCI-compatible host bridge from the memory address range allocated to the PCI-compatible host bridge.

27. (Original) The apparatus of claim 26, wherein the program code is configured to non-uniformly allocate memory address ranges to the plurality of IO slots by allocating a remaining memory address range to a last IO slot among the plurality of IO slots, wherein the remaining memory address range comprises that portion of the memory address range allocated to the PCI-compatible host that was not allocated to each other IO slot among the plurality of IO slots.

28. (Original) The apparatus of claim 15, wherein the IO fabric includes at least one IO fabric element allowing connectivity to a subset of the plurality of IO endpoints, and wherein the memory address ranges that would be allocated to IO resources coupled to each IO endpoint in the plurality of IO endpoints are determinable from a publication available prior to installation of IO resources in the subset of IO endpoints.

29. (Original) The apparatus of claim 15, further comprising the plurality of IO resources and the memory mapped IO fabric.

30. (Currently Amended) A program product, comprising:

program code configured to allocate a memory address space to a plurality of input/output (IO) resources coupled to a plurality of IO endpoints in a memory mapped IO fabric by determining a location in the memory mapped IO fabric for each IO endpoint among the plurality of endpoints, and non-uniformly allocating memory address ranges to the plurality of IO endpoints based upon the determined locations of the IO endpoints in the memory mapped IO fabric, wherein the program code is configured to allocate memory address ranges to the plurality of IO endpoints by allocating differently sized memory address ranges to first and second IO endpoints having the same connector type; and

a tangible computer readable medium bearing the program code.

31. (Previously Presented) The program product of claim 30, wherein the computer readable medium includes a recordable medium.

32. (New) A method of allocating memory addresses to a plurality of input/output (IO) resources coupled to a plurality of IO endpoints in a memory mapped IO fabric, the method comprising:

determining a location in the memory mapped IO fabric for each IO endpoint among the plurality of endpoints; and

non-uniformly allocating memory address ranges to the plurality of IO endpoints based upon the determined locations of the IO endpoints in the memory mapped IO fabric;

wherein determining the location and non-uniformly allocating memory address ranges are performed during initialization of the memory mapped IO fabric, wherein determining the location and non-uniformly allocating memory address ranges are performed during initialization of a computer to which the memory mapped IO fabric is coupled, wherein the computer comprises a logically-partitioned computer, and wherein determining the location and non-uniformly allocating memory address ranges are performed by a partition manager in the logically-partitioned computer.

33. (New) A method of allocating memory addresses to a plurality of input/output (IO) resources coupled to a plurality of IO endpoints in a memory mapped IO fabric, the method comprising:

determining a location in the memory mapped IO fabric for each IO endpoint among the plurality of endpoints; and

non-uniformly allocating memory address ranges to the plurality of IO endpoints based upon the determined locations of the IO endpoints in the memory mapped IO fabric;

wherein the memory mapped IO fabric comprises a PCI-compatible fabric, wherein the memory mapped IO fabric comprises a plurality of PCI-compatible buses, wherein at least a subset of IO endpoints are IO slots coupled to the plurality of PCI-compatible buses, wherein the location of each IO slot is defined by a bus identifier for the PCI-compatible bus to which such IO slot is coupled, and a slot identifier for such IO slot on

the PCI-compatible bus to which such IO slot is coupled, wherein a PCI-compatible bus among the plurality of PCI-compatible buses includes a PCI-compatible host bridge, the method further comprising allocating a memory address range to the PCI-compatible host bridge, wherein non-uniformly allocating memory address ranges to the plurality of IO endpoints includes allocating memory address ranges to a plurality of IO slots coupled to the PCI-compatible host bridge from the memory address range allocated to the PCI-compatible host bridge.

34. (New) An apparatus, comprising:

at least one processor;

program code configured to be executed by the at least one processor to allocate memory address space to a plurality of input/output (IO) resources coupled to a plurality of IO endpoints in a memory mapped IO fabric by determining a location in the memory mapped IO fabric for each IO endpoint among the plurality of endpoints, and non-uniformly allocating memory address ranges to the plurality of IO endpoints based upon the determined locations of the IO endpoints in the memory mapped IO fabric, wherein the program code is configured to determine the location and non-uniformly allocate memory address ranges during initialization of the memory mapped IO fabric, wherein the program code is configured to determine the location and non-uniformly allocate memory address ranges during initialization of the apparatus, wherein the apparatus comprises a logically-partitioned computer, and wherein the program code is resident in a partition manager in the logically-partitioned computer.

35. (New) An apparatus, comprising:

at least one processor;

program code configured to be executed by the at least one processor to allocate memory address space to a plurality of input/output (IO) resources coupled to a plurality of IO endpoints in a memory mapped IO fabric by determining a location in the memory mapped IO fabric for each IO endpoint

among the plurality of endpoints, and non-uniformly allocating memory address ranges to the plurality of IO endpoints based upon the determined locations of the IO endpoints in the memory mapped IO fabric, wherein the memory mapped IO fabric comprises a PCI-compatible fabric, wherein the memory mapped IO fabric comprises a plurality of PCI-compatible buses, wherein at least a subset of IO endpoints are IO slots coupled to the plurality of PCI-compatible buses, wherein the location of each IO slot is defined by a bus identifier for the PCI-compatible bus to which such IO slot is coupled, and a slot identifier for such IO slot on the PCI-compatible bus to which such IO slot is coupled, wherein a PCI-compatible bus among the plurality of PCI-compatible buses includes a PCI-compatible host bridge, wherein the program code is further configured to allocate a memory address range to the PCI-compatible host bridge, and wherein the program code is configured to non-uniformly allocate memory address ranges to the plurality of IO endpoints by allocating memory address ranges to a plurality of IO slots coupled to the PCI-compatible host bridge from the memory address range allocated to the PCI-compatible host bridge.